



## Applesoft BASIC Quick Reference Card

### Editing and Cursor Control

[LEFT-ARROW]  
 [RIGHT-ARROW]  
 [CONTROL]-X

Erase previous character  
 Recopy character under cursor  
 Cancel input line

[ESC] A  
 [ESC] B  
 [ESC] C  
 [ESC] D

Move right; leave escape mode  
 Move left; leave escape mode  
 Move down; leave escape mode  
 Move up; leave escape mode

[ESC] I

Move up; remain in escape mode

[ESC] J

Move left; remain in escape mode

[ESC] K

Move right; remain in escape mode

[ESC] M

Move down; remain in escape mode

After [ESC], arrow keys are the same as I, J, K, M

[ESC] E  
 [ESC] F  
 [ESC] @

Clear to end of line  
 Clear to end of screen  
 Clear entire screen; move cursor to top

DEL *n1* ; *n2*

Delete from line *n1* to line *n2*

### Statements and Lines

Lines typed without a line number are executed immediately; those with a line number are saved for later (deferred) execution.

: Separates multiple statements on same line

REM Remarks for human reader

### Operations on Whole Programs

NEW

Erase current program, reset all variables

CLEAR

Reset all variables

LIST

Display current program

LIST *n1* - *n2*

Display from line *n1* to line *n2*

RUN

Execute program from beginning

RUN *n*

Execute program starting at line *n*

RUN *name*

Load and execute program name from disk

LOAD

Load program from tape

LOAD *name*

Load program name from disk

SAVE

Save current program on tape

SAVE *name*

Save current program on disk as *name*

### Interrupting and Resuming

[CONTROL]-S

Suspend output (any key to resume)

[CONTROL]-C

Interrupt program execution

CONT

Continue execution after

[CONTROL]-C, STOP, or END

[CONTROL]-[RESET]

Cancel program execution

### Variables

Type Name Range

Real AB + / - 9.999999999 E + 37

Integer AB% + / - 32767

String AB\$ 0 to 255 characters

where A is a letter, B is a letter or digit. Name may be more than two characters, but only first two are significant.

## Control

GOTO <i>n</i>	Branch to line <i>n</i>
ON <i>expr</i> GOTO <i>n1</i> , <i>n2</i> , <i>n3</i> , ...	Branch to line <i>n1</i> , <i>n2</i> , <i>n3</i> , ... depending on value of <i>expr</i>
IF <i>cond</i> THEN <i>s1</i> : <i>s2</i> : <i>s3</i> : ...	Execute statements <i>s1</i> , <i>s2</i> , <i>s3</i> , ... if condition <i>cond</i> is true
FOR <i>v</i> = <i>x</i> TO <i>y</i> STEP <i>z</i>	Begin loop for all values of <i>v</i> from <i>x</i> to <i>y</i> by <i>z</i> ; if STEP omitted, 1 is understood
NEXT <i>v</i>	Repeat loop for next value of <i>v</i>
GOSUB <i>n</i>	Branch to subroutine at line <i>n</i>
RETURN	Return from subroutine to point of call
ON <i>expr</i> GOSUB <i>n1</i> , <i>n2</i> , <i>n3</i> , ...	Branch to subroutine at line <i>n1</i> , <i>n2</i> , <i>n3</i> , ... depending on value of <i>expr</i>
POP	Remove last return address from subroutine stack without branching
ONERR GOTO <i>n</i>	Establish error-handling routine beginning at line <i>n</i>
RESUME	Reexecute statement causing error
STOP	Halt execution with message identifying line
END	Halt execution with no message

## String Operations

+	Concatenate strings
LEN ( <i>s</i> )	Length of string <i>s</i>
LEFT\$ ( <i>s</i> , <i>x</i> )	Leftmost <i>x</i> characters of string <i>s</i>
MID\$ ( <i>s</i> , <i>x</i> , <i>y</i> )	<i>y</i> characters beginning at position <i>x</i> in string <i>s</i>
RIGHT\$ ( <i>s</i> , <i>x</i> )	Rightmost <i>x</i> characters of string <i>s</i>
STR\$ ( <i>x</i> )	String representing numeric value <i>x</i>
VAL ( <i>s</i> )	Numeric value of string <i>s</i>
CHR\$ ( <i>x</i> )	Character with ASCII code <i>x</i>
ASC ( <i>s</i> )	ASCII code for first character of string <i>s</i>

## Input/Output

IN# <i>n</i>	Accept input from slot <i>n</i>
IN# 0	Accept input from keyboard
INPUT <i>s</i> ; <i>x</i> , <i>y</i> , <i>z</i>	Prompt with string <i>s</i> , then read values into variables <i>x</i> , <i>y</i> , <i>z</i> ; if <i>s</i> omitted, ? is used
GET <i>c</i>	Read one character into variable <i>c</i>
READ <i>x</i> , <i>y</i> , <i>z</i>	Read values from DATA list into variables <i>x</i> , <i>y</i> , <i>z</i>
DATA <i>x</i> , <i>y</i> , <i>z</i>	Add values <i>x</i> , <i>y</i> , <i>z</i> to DATA list
RESTORE	Restart DATA list from beginning
RECALL <i>a</i>	Read array <i>a</i> from tape
PDL ( <i>n</i> )	Read dial of hand control <i>n</i>
PR# <i>n</i>	Send output to slot <i>n</i>
PR# 0	Send output to display screen
PRINT <i>x</i> , <i>y</i> , <i>z</i>	Display or print values <i>x</i> , <i>y</i> , <i>z</i>
STORE <i>a</i>	Write array <i>a</i> to tape
TEXT	Display text
HOME	Clear screen and send cursor to top
;	Start next item at cursor position
,	Start next item at next tab position
SPC ( <i>x</i> )	Display or print <i>x</i> spaces (PRINT statement only)
TAB ( <i>x</i> )	Move cursor to column <i>x</i> (PRINT statement only)
HTAB <i>x</i>	Move cursor to column <i>x</i>
VTAB <i>y</i>	Move cursor to line <i>y</i>
POS (0)	Current horizontal cursor position
INVERSE	Display text in black-on-white
FLASH	Display flashing text
NORMAL	Display text in white-on-black
SPEED = <i>x</i>	Set text display rate to <i>x</i> (0 minimum, 255 maximum)



## Arrays

Type	Typical Element
Real	AB (x, y, z)
Integer	AB% (x, y, z)
String	AB\$ (x, y, z)

where A is a letter, B is a letter or digit. Name may be more than two characters, but only first two are significant. Array size limited only by available memory.

**DIM a (x, y, z)** Define array a with maximum subscripts x, y, z

## Arithmetic Operators

=	Assign value to variable (LET optional)
+	Addition
-	Subtraction
*	
/	Division
^	Exponentiation

## Relational Operators

=	Equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
<>	Not equal to

Yield value 1 if true, 0 if false. Can also be used to compare strings.

## Logical Operators

AND	Both true
OR	Either or both true
NOT	Is false

Interpret 0 as false, nonzero as true. Yield value 0 if false, 1 if true.

## Precedence of Operators

( )	Parentheses (innermost first)
+ - NOT	Signed arithmetic, logical "not"
^	Exponentiation
* /	Multiplication, division
+ -	Addition, Subtraction
= < >	Relational operators
<= =<	
>= =>	
<> ><	
AND	Logical "and"
OR	Logical "or"

## Arithmetic Functions

ABS (x)	Absolute value of x
SGN (x)	Sign of x
INT (x)	Integer part of x
SQR (x)	Square root of x
SIN (x)	Sine of x radians
COS (x)	Cosine of x radians
TAN (x)	Tangent of x radians
ATN (x)	Arc tangent, in radians, of x
EXP (x)	Exponential of x
LOG (x)	Natural logarithm of x
RND (x)	If x > 0, generate random number between 0 and 1 If x = 0, repeat previous random number If x < 0, begin new repeatable sequence of random numbers
DEF FN (x)	Define function
= expr	

## Graphics

GR	Display low-resolution graphics
COLOR = <i>x</i>	Set low-resolution display color to <i>x</i>
PLOT <i>x</i> , <i>y</i>	Plot single block at column <i>x</i> , row <i>y</i>
HLIN <i>x1</i> , <i>x2</i> AT <i>y</i>	Draw horizontal line from column <i>x1</i> to column <i>x2</i> in row <i>y</i>
VLIN <i>y1</i> , <i>y2</i> AT <i>x</i>	Draw vertical line from row <i>y1</i> to row <i>y2</i> in column <i>x</i>
SCRN ( <i>x</i> , <i>y</i> )	Color on screen at column <i>x</i> , row <i>y</i>

Columns numbered from 0 to 39; rows from 0 to 39 in mixed text and graphics, 0 to 47 in full-screen graphics.

HGR	Display high-resolution graphics, page 1; mixed text and graphics
HGR2	Display high-resolution graphics, page 2; full-screen graphics
HCOLOR = <i>x</i>	Set high-resolution display color to <i>x</i>
HPOINT <i>x</i> , <i>y</i>	Plot single point at column <i>x</i> , row <i>y</i>
HPOINT <i>x1</i> , <i>y1</i> TO <i>x2</i> , <i>y2</i> TO <i>x3</i> , <i>y3</i>	Draw high-resolution lines from column <i>x1</i> , row <i>y1</i> to column <i>x2</i> , row <i>y2</i> to column <i>x3</i> , row <i>y3</i>
HPOINT TO <i>x</i> , <i>y</i>	Extend previous line to column <i>x</i> , row <i>y</i>

Columns numbered from 0 to 279; rows from 0 to 159 in mixed text and graphics, 0 to 191 in full-screen graphics.

SHLOAD	Load shape table from tape
DRAW <i>n</i> AT <i>x</i> , <i>y</i>	Draw shape number <i>n</i> at column <i>x</i> , row <i>y</i>
XDRAW <i>n</i> AT <i>x</i> , <i>y</i>	Erase shape number <i>n</i> at column <i>x</i> , row <i>y</i>
SCALE = <i>x</i>	Set scale factor for drawing shapes to <i>x</i>
ROT = <i>x</i>	Set rotation for drawing shapes to <i>x</i>

## Utility Statements

PEEK ( <i>addr</i> )	Contents of memory location <i>addr</i>
POKE <i>addr</i> , <i>x</i>	Store value <i>x</i> at memory location <i>addr</i>
CALL <i>addr</i>	Execute machine-language subroutine starting at location <i>addr</i>
USR ( <i>x</i> )	Execute user-supplied machine-language function routine with argument <i>x</i>
WAIT <i>addr</i> , <i>m1</i> , <i>m2</i>	Suspend execution until bit pattern specified by masks <i>m1</i> , <i>m2</i> appears at location <i>addr</i>
HIMEM: <i>addr</i>	Set highest memory address available for variable storage to <i>addr</i>
LOMEM: <i>addr</i>	Set lowest memory address available for variable storage to <i>addr</i>
FRE (0)	Amount of available storage remaining
TRACE	Display line number of each statement executed
NOTRACE	Stop displaying line number of each statement executed